BAB II

LANDASAN TEORI

Landasan teori merupakan dasar atau acuan bagi peneliti dalam menyelesaikan rumusan masalah dengan metode-metode yang akan dipaparkan dalam pembahasan untuk ditarik kesimpulan.

2.1. Distribusi dan Transpotrasi

Menurut Prihatinie (2011) Distribusi merupakan suatu proses pengiriman barang dari depot ke *costumer* dalam upaya mencapai keberhasilan penjualan dan kepuasan *costumer* persoalan distribusi sangat penting karena berhubungan dengan biaya transportasi yang mempengaruhi biaya produksi. Secara umum fungsi distribusi dan transportasi pada dasarnya adalah mengantarkan produk dari lokasi dimana produk tersebut diproduksi sampai dimana mereka akan digunakan. Manajemen transportasi dan distribusi mencangkup baik aktivitas fisik yang secara kasat mata bisa kita saksikan, seperti menyimpan dan mengirim produk, maupun fungsi non fisik yang berupa aktivitas pengolahan informasi dan pelayanan kepada pelanggan (Pujawan 2010). Pada prinsipnya, fungsi ini bertujuan untuk menciptakan pelayanan yang tinggi ke pelanggan yang bisa dilihat dari tingkat *service level* yang dicapai, kecepatan pengiriman, kesempurnaan barang sampai ketangan pelanggan, serta pelayanan purna jurnal yang memuaskan. Kurang baiknya perencanaan system distribusi akan mengarah pada pemborosan biaya transportasi dan penurunan kepuasan konsumen yang selnjutnya menyebabkan hilangnya kepercayaan (Ikfan 2013).

Kegiatan transportasi dan distribusi bisa dilakukan oleh perusahaan manufaktur dengan membentuk bagian distribusi / transportasi tersendiri atau diserahkan kepada pihak ketiga. Dalam upayanya untuk memenuhi tujuan – tujuan diatas, siapapun yang melaksanakan (internal perusahaan atau mitra pihak ketiga), manajemen distribusi dan transportasi pada umumnya melakukan sejumlah fungsi dasar yang terdiri dari :

1. Melakukan segmentasi dan menentukan target *service level*. Segmentasi pelanggan perlu dilakukan karena kontribusi mereka pada *revenue* perusahaan bisa sangat bervariasi dan karakteristik tiap pelanggan bisa sangat berbeda antara satu dengan yang lainnya.
2. Menentukan mode transportasi yang akan digunakan. tiap metode transportasi memiliki karakteristik yang berbeda dan mempunyai keunggulan serta kelemahan yang berbeda juga. Sebagai contoh, transportasi laut memiliki keunggulan dari segi biaya yang lebih rendah, namun lebih lambat dibandingkan dengan transportasi udara. Manajemen transportasi harus bisa menentukan mode apa yang akan digunakan dalam mengirimkan / mendistribusikan produk-produk mereka ke pelanggan. Kombinasi dua atau lebih mode transportasi tentu bisa atau bahkan harus dilakukan tergantung pada situasi yang dihadapi.
3. Melakukan konsolidasi informasi dan pengiriman. Konsolidasi merupakan kata kunci yang sangat penting, tekanan untuk melakukan pengiriman cepat namun murah menjadi pendorong utama perlunya melakukan konsolidasi informasi maupun pengiriman.
4. Melakukan penjadwalan dan penentuan rute pengiriman. Salah satu kegiatan operasional yang dilakukan oleh gudang atau distributor adalah menentukan kapan sebuah truk harus berangkat dan rute mana yang harus dilalui untuk memenuhi permintaan dari sejumlah pelanggan. Apabila jumlah pelanggan sedikit, keputusan ini bisa diambil dengan *relatife* gampang. Namun perusahaan yang memiliki pulhan bahkan ratusan toko untuk dikunjungi, penjadwalan dan penentuan rute pengiriman adalah pekerjaan yang sangat sulit dan kekurangtepatan dalam mengambil dua keputusan bisa berimplikasi pada biaya pengiriman dan penyimpanan yang tinggi.
5. Memberikan pelayanan nilai tambah. Disamping mengirimkan produk ke pelanggan, jaringan distribusi semakin banyak dipercaya untuk melakukan proses nilai tambah. Beberapa proses nilai tambah yang bisa dikerjakan oleh distributor adalah pengepakan, pelabelan harga, pemberian *barcode*, dan sebagainya.

6. Menyimpan persediaan. Jaringan distribusi selalu melibatkan proses penyimpanan produk baik disuatu gudang pusat atau gudang *regional* maupun ditoko dimana produk tesebut dipajang untuk dijual. Oleh karena itu manajemen ditribusi tidak bisa dilepaskan dari manajemen pergudangan.
7. Menangani pengembalian *(return)*. Manajemen distribusi juga punya tanggung jawab untuk melaksanakan kegiatan pengembalian produk dari hilir ke hulu dalam *supply chain*. Pengembalian ini bisa karena produk rusak atau tidak terjual sampai batas waktu penjualan habis, seperti produk makanan, sayur, buah, dan sebagainya.

Menurut Pujawan (2010) Secara umum ada tiga strategi distribusi produk dari pabrik ke pelanggan. Masing-masing dari strategi ini memiliki kelebihan dan kekurangan. Ketiga strategi tersebut adalah sebagai berikut:

1. Pengiriman Langsung *(Direct Shipment)*

   Biasanya strategi ini cocok digunakan untuk barang yang umurnya pendek dan barang yang mudah rusak dalam proses bongkar/muat atau pemindahan. Karena hilangnya fasilitas antara (gudang), maka ada penghematan biaya fasilitas, tetapi terkadang biaya transportasi lebih tinggi akibat berkurangnya kesempatan mencapai *economies of scale* yang tinggi pada aktifitas transportasi. Keunggulan lainnya adalah pemendekan waktu kirim dari pabrik ke pelanggan dan pengurangan *inventory* pada *supply chain*. Di sisi lain, strategi ini akan menanggung risiko yang lebih tinggi bila ketidakpastian permintaan maupun ketidakpastian pasokan relatif tinggi.
2. Pengiriman Melalui *Warehouse*

   Berkebalikan dengan model *direct shipment*, model *warehousing* cocok untuk produk produk yang ketidakpastian *demand / supplynya* tinggi serta produk-produk yang memiliki daya tahan relatif lama *(durable products)*. Gudang juga berfungsi sebagai tempat melakukan konsolidasi beban dari sejumlah supplier ke sejumlah pelanggan sehingga pengiriman bisa dilaksanakan dengan skala ekonomi yang lebih tinggi.

3. *Cross - Docking*

   Di tempat ini, kendaraan penjemput dan pengirim akan bertemu dan terjadi transfer beban (tentu juga dimungkinkan terjadinya konsolidasi yang melibatkan banyak pabrik dan pelanggan). Secara umum keunggulannya adalah pengiriman bisa relatif cepat dan tetap bisa mencapai economies of transportation yang baik karena adanya konsolidasi. Disamping itu, kegiatan handling akan jauh berkurang dan *inventory* di *supply chain* tidak akan setinggi *model warehousing*

2.2. *Vehicle Routing Problem (VRP)*

Menurut Ikfan (2013) *Vehicle Routing Problem (VRP)* merupakan penentuan sejumlah rute untuk sekumpulan kendaraan yang harus dilayani sejumlah pemberhentian (node) dari depot pusat. *VRP* dapat didefinisikan sebagai permasalahan pencairan rute distribusi dengan ongkos minimal dari satu depot ke pelanggan yang letaknya tersebar dengan jumlah permintaan yang berbeda – beda (Arvianto 2014). Asumsi yang biasa digunakan untuk *Vehicle Routing Problem* adalah setiap kendaraan mempunyai kapasitas yang sama, jumlah kendaraan tidak terbatas, jumlah permintaan tiap pemberhentian (node) diketahui dan tidak ada jumlah permintaan tunggal yang melebihi kapasitas kendaraan. *Vehicle Routing Problem (VRP)* diperkenalkan pertama kali oleh Dantziq dan Ramser pada tahun 1959 dan semenjak itu telah dipelajari secara luas. Oleh Fisher, VRP didefinisikan sebagai sebuah pencarian atas cara penggunaan yang efisien dari sejumlah *vehicle* yang harus melakukan perjalanan untuk mengunjungi sejumlah tempat untuk mengantar dan / atau menjemput orang / barang. Istilah customer digunakan untuk menunjukkan pemberhentian untuk mengantar dan / atau menjemput orang / barang. Setiap *customer* harus dilayani oleh satu *vehicle* saja. Penentuan pasangan *vehicle-customer* ini dilakukan dengan mempertimbangkan kapasitas *vehicle* dalam satu kali angkut, untuk meminimalkan biaya yang diperlukan. Biasanya, penentuan biaya minimal erat kaitannya dengan jarak yang minimal (Christian, 2011).

Menurut Christian (2011) VRP juga dapat dilihat sebagai kombinasi dari dua permasalahan optimasi lain, yaitu *Bin Packing Problem (BPP)* dan *Travelling*

Salesman Problem (TSP). BPP dapat dideskripsikan sebagai berikut: “Diberikan sejumlah angka, yang melambangkan ukuran dari sejumlah item, dan sebuah konstanta K, yang melambangkan kapasitas dari bin. Berapa jumlah bin minimum yang diperlukan?” Tentu saja satu item hanya dapat berada dalam satu bin saja, dan total kapasitas item pada setiap bin tidak boleh melebihi kapasitas dari bin tersebut. Di samping itu, TSP adalah sebuah permasalahan tentang seorang salesman yang ingin mengunjungi sejumlah kota. Dia harus mengunjungi tiap kota sekali saja, dimulai dan diakhiri dari kota awal. Inti permasalahan adalah untuk menemukan jalur terpendek melalui semua kota yang ada. Hubungan keduanya dengan VRP adalah, *vehicle* dapat dihubungkan dengan customer menggunakan BPP, dan urutan kunjungan vehicle terhadap tiap customer diselesaikan menggunakan TSP.

2.3. Klasifikasi VRP

Menurut Rohandi (2014) Terdapat beberapa jenis VRP yang sangat bergantung pada jumlah faktor pembatas dan tujuan yang akan dicapai. Pembatas yang paling umum digunakan yaitu waktu dan jarak. Tujuan yang ingin dicapai biasanya minimasi, waktu tempuh, waktu maupun biaya. Beberapa contoh variasi VRP diantaranya:

Variasi bentuk VRP muncul pada suatu kondisi yang ada. Kondisi terdiri dari sejumlah faktor, kendala, dan fungsi tujuan. beberapa contoh variasi dari VRP, antara lain:

1. CVRP: salah satu jenis dari *vehicle routing problem* yang dibatasi oleh kapasitas kendaraan.
2. *VRP with multiple trips*: satu kendaraan dapat melakukan lebih dari satu rute untuk memenuhi kebutuhan pelanggan.
3. *VRP with time window*: setiap pelanggan mempunyai rentang waktu pelayanan yaitu pelayanan harus dilakukan pada rentang *time window* masing-masing pelanggan.
4. *VRP with split deliveries*: setiap pelanggan boleh dikunjungi lebih dari satu kendaraan.

5. *VRP with multiple products*: permintaan pelanggan lebih dari satu produk. Pada umumnya, VRP bentuk ini juga melibatkan kendaraan dengan *multi - compartments.*
6. *Periodic VRP*: adanya horison perencanaan yang berlaku untuk satuan waktu tertentu.
7. *VRP with delivery dan pick - up*: terdapat sejumlah barang yang perlu dipindahkan dari lokasi penjemputan tertentu ke lokasi pengiriman lainnya.
8. *VRP with multiple depots*: depot awal untuk melayani pelanggan lebih dari satu.
9. *VRP with heterogeneous fleet of vehicle*: kapasitas kendaraan antara kendaraan satu dengan kendaraan lain. Jumlah dan tipe kendaraan diketahui.
10. *Stochastic VRP*: memiliki unsur random misalnya permintaan pelanggan yang tidak pasti dan waktu perjalanan
11. *Dynamic VRP*: pelanggan baru dapat disisipkan pada perencanan rute selanjutnya.

2.4. *Capacitated Vehicle Routing Problem* (CVRP)

Menurut Sulistiono (2015) *Vehicle Routing Problem (VRP)* merupakan bagian dari TSP, artinya VRP merupakan TSP dengan menyertakan kendala satu kendaraan dengan kapasitas. Beberapa komponen beserta karakteristiknya yang terdapat dalam masalah VRP menurut Toth dan Vigo (2002), yaitu depot, jaringan jalan, konsumen, kendaraan dan pengemudi

Vehicle Routing Problem pertama kali diperkenalkan oleh Damtizig dan Ramser pada tahun 1959. VRP sebenarnya merupakan perkembangan atau perluasan dari Travel Salesman Problem (TSP). versi yang paling dasar dari VRP adalah Capatitated Vehicle Routing Problem (CVRP) (Tanujaya 2011).

Capacitated Vehicle Routing Problem (CVRP) merupakan salah satu variasi dari masalah VRP dengan kendala kapasitas kendaraan yang terbatas. CVRP dapat direpresentasikan sebagai suatu *graf* berarah berbobot *(weighted directed graph)* D

= (V,A) dimana V = $\{v_0, v_1, v_2, \ldots, v_n\}$ adalah himpunan simpul *(nodes)* dan $A=\{(v_i,v_j)|v_i,v_j \in V, i \neq j\}$ adalah himpunan busur *(arcs)* yang menghubungkan himpunan simpul *(nodes)*. Simpul dinyatakan sebagai depot dan yang lainnya adalah pelanggan. Setiap elemen dari busur *(arcs)* menyatakan jarak. Sedangkan setiap simpul memiliki permintaan *(demand)* yang dinotasikan sebagai dengan i= 1,2,3,… n. Himpunan $K= \{k_1, k_2, k_3,\ldots, k_n\}$ merupakan kumpulan kendaraan yang homogen. Kapasitas kendaraan yang digunakan dinotasikan dengan Q[3].

Diberikan i j d adalah jarak dari simpul i ke simpul j (d_{ij} merupakan bilangan nonnegative). Jarak diasumsikan semetrik ($d_{ij} = d_{ji}$) dan ($d_{ii} = d_{jj} = 0$). Permasalahan tersebut kemudian dapat dibuat menjadi model matematika dengan tujuan meminimumkan total jarak tempuh perjalanan kendaraan:

Didefinisikan variabel keputusan

$x^k_{i,j}$
- 1, jika kendaraan k mengunjungi simpul v_j setelah simpul v_i
- 0, selainnya

y^k_i
- 1, jika simpul v_i dilayani oleh kendaraan k
- 0, selainnya

Keterangan variable

- D = (V,A)
- V = himpunan simpul, $\{v_0, v_1, v_2, \ldots, v_n\}$ dimana v_0 adalah depot dan $v_1, v_2, \ldots, v_n$ adalah pelanggan
- A = himpunan sisi berarah *(arcs)*, $\{(v_i,v_j)|v_i,v_j \in V, i \neq j\}$
- D_{ij} = jarak antara simpul v_i ke simpul v_j
- q_1 = permintaan pelanggan ke i, i € V
- K = { $k_1, k_2, k_3, \ldots, k_n$} kendaraan seragam yang digunakan □
- Q adalah kapasitas masing-masing kendaraan , k_i € K, i={1,2,3,…, n}

Fungsi tujuannya meminimumkan total jarak tempuh kendaraan. Jika Z adalah fungsi tujuan, maka

$$\text{Min } Z = \sum_{k \in K} ij \sum_{i \in V} ij \sum_{j \in V} d\ x^k \qquad (1)$$

Kendala

a. Setiap simpul hanya boleh dikunjungi tepat satu kali oleh 1 kendaraan.

$$\sum_{k \in K} \sum_{j \in N} x_{i,j} = 1,\ \forall i \in V \qquad (2)$$

b. Kendaraan yang telah mengunjungi simpul *i,* kendaraan *k* harus meninggalkan simpul tersebut menuju simpul lain.

$$\sum_{j \in V} x^k_{i,j} - \sum_{k} x_{j,i} = 0,\ \forall i \in V, \forall k \in K \qquad (3)$$

$$j \in V$$

c. Total jumlah permintaan pelanggan dalam satu rute tidak melebihi kapasitas kendaraan.

$$\sum_{i \in V} q_i \sum_{j, i \in V, j \neq i} x^k_{xij} \leq Q, \forall k \in K \qquad (4)$$

d. Setiap rute perjalanan kendaraan berawal dari depot

$$\sum_{j \in V} x_{0,j} = 1, \forall k \in K \quad (5)$$

e. Setiap rute perjalanan kendaraan berakhir di depot

$$\sum_{j \in V} x_{j,0} = 1, \forall k \in K \quad (6)$$

f. Batasan ini memastikan bahwa tidak terdapat subrute pada setiap rute yang terbentuk.

$$x_{i,j}^{k} = 1 \Rightarrow y_i - q_j = y_j, \forall i, j \in V, k \in K \quad (7)$$

$$y_0 = Q, 0 \leq y_i \forall i \in V \quad (8)$$

g. Variable keputusan $x_{i,j}^{k}$ merupakan bilangan *biner*

$$x_{i,j}^{k} \in \{0,1\}, \forall i, j \in V, k \quad (9)$$

Variabel keputusan hanya akan terdefinisi jika jumlah permintaan simpul dari simpul tidak melebihi kapasitas kendaraan. Apabila kapasitas kendaraan tidak memadahi untuk pelanggan berikutnya maka kendaraan harus mengisi muatan di depot sehingga akan terbentuk rute baru.

Variabel keputusan hanya akan terdefinisi jika jumlah permintaan simpul dari simpul tidak melebihi kapasitas kendaraan. Apabila kapasitas kendaraan tidak memadahi untuk pelanggan berikutnya maka kendaraan harus mengisi muatan di depot sehingga akan terbentuk rute baru.

2.5. Algoritma Penghematan (*SavingAlgorithm*)

Menurut Christian (2011) Pada tahun 1964, Clarke dan Wright mempublikasikan sebuah algoritma sebagai solusi permasalahan dari berbagai rute kendaraan, yang sering disebut sebagai permasalahan klasik dari rute kendaraan

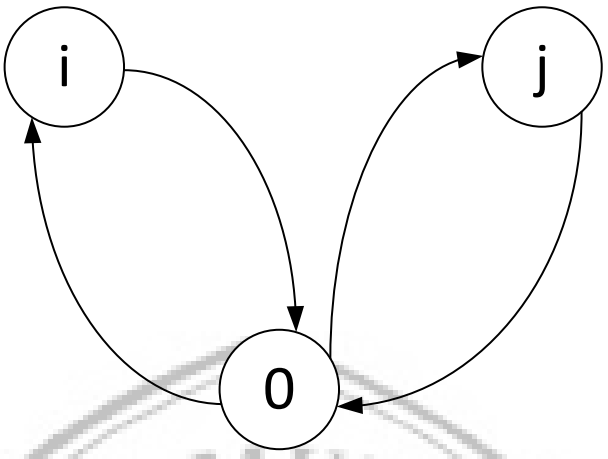
(the classical vehicle routing problem). Algoritma ini didasari pada suatu konsep yang disebut konsep *savings*.

Algoritma ini dirancang untuk menyelesaikan masalah rute kendaraan dengan karakteristik sebagai berikut. Dari suatu depot barang harus diantarkan kepada pelanggan yang telah memesan. Untuk sarana transportasi dari barang barang ini, sejumlah kendaraan telah disediakan, di mana masing-masing kendaraan dengan kapasitas tertentu sesuai dengan barang yang diangkut. Setiap kendaraan yang digunakan untuk memecahkan permasalahan ini, harus menempuh rute yang telah ditentukan, memulai dan mengakhiri di depot, di mana barang-barang diantarkan kepada satu atau lebih pelanggan

Permasalahannya adalah untuk menetapkan alokasi untuk pelanggan di antara rute-rute yang ada, urutan rute yang dapat mengunjungi semua pelanggan dari rute yang ditetapkan dari kendaraan yang dapat melalui semua rute. Tujuannya adalah untuk menemukan suatu solusi yang meminimalkan total pembiayaan kendaraan. Lebih dari itu, solusi ini harus memuaskan batasan bahwa setiap pelanggan dikunjungi sekali, di mana jumlah yang diminta diantarkan, dan total permintaan pada setiap rute harus sesuai dengan kapasitas kendaraan.

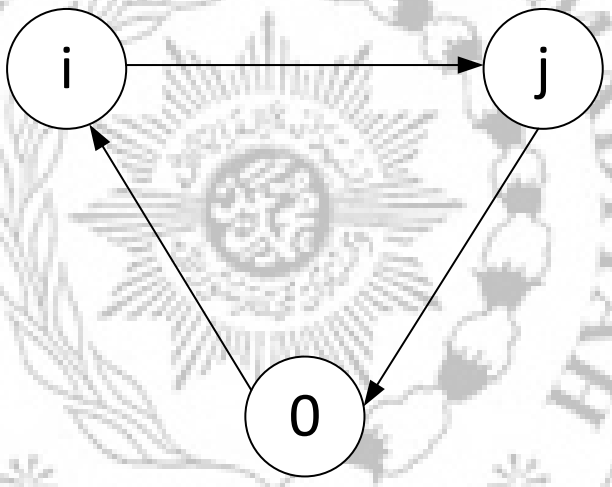
Biaya-biaya kendaraan ditetapkan oleh biaya pengangkutan dari beberapa titik ke titik-titik yang lain. Pembiayaan tidak harus sama pada dua jalur di antara dua titik. Metode Saving Matriks merupakan metode yang dapat digunakan untuk menetukan pengelompokan atau penggabungan dua atau lebih lokasi/costumer ke dalam suatu armada. Dengan memperhatikan penghematan jarak dan kapasitas armada yang digunakan.

Menurut Christian (2011) Algoritma *savings* adalah sebuah algoritma heuristik, dan oleh karena itu tidak menyediakan sebuah solusi yang optimal untuk problem tertentu. Metode ini, bagaimanapun juga sering menghasilkan solusi yang baik. Yang merupakan suatu solusi yang sedikit berbeda dari solusi optimal. Dasar dari konsep penghematan ini untuk mendapatkan penghematan biaya dengan menggabungkan dua rute menjadi satu rute yang digambarkan pada Gambar 2.1 A dan 2.1 B , titik 0 adalah depot.



Gambar 2.1 A Ilustrasi Konsep Penghematan

Sumber: Jens Lysgaard (2007)



Gambar 2.1 B Ilustrasi Konsep Penghematan

Sumber: Jens Lysgaard (2007)

Berdasarkan Gambar 2.1 A pelanggan i dan j dikunjungi dengan rute yang terpisah. Sebuah alternatif untuk masalah ini adalah mengunjungi dua pelanggan pada rute yang sama, sebagai contoh pada urutan i – j seperti yang diperlihatkan pada Gambar 2.1 B. karena biaya transportasi diberikan, penghematan yang terjadi dari pengangkutan pada rute Gambar 2.1 B dibanding dua rute pada Gambar 2.1 A dapat dihitung. Biaya kendaraan yang ditunjukkan di antara titik i dan j oleh cij, total biaya kendaraan oleh Da pada Gambar 2.1 A adalah:

Da = c0i + ci0 + c0j + cj0

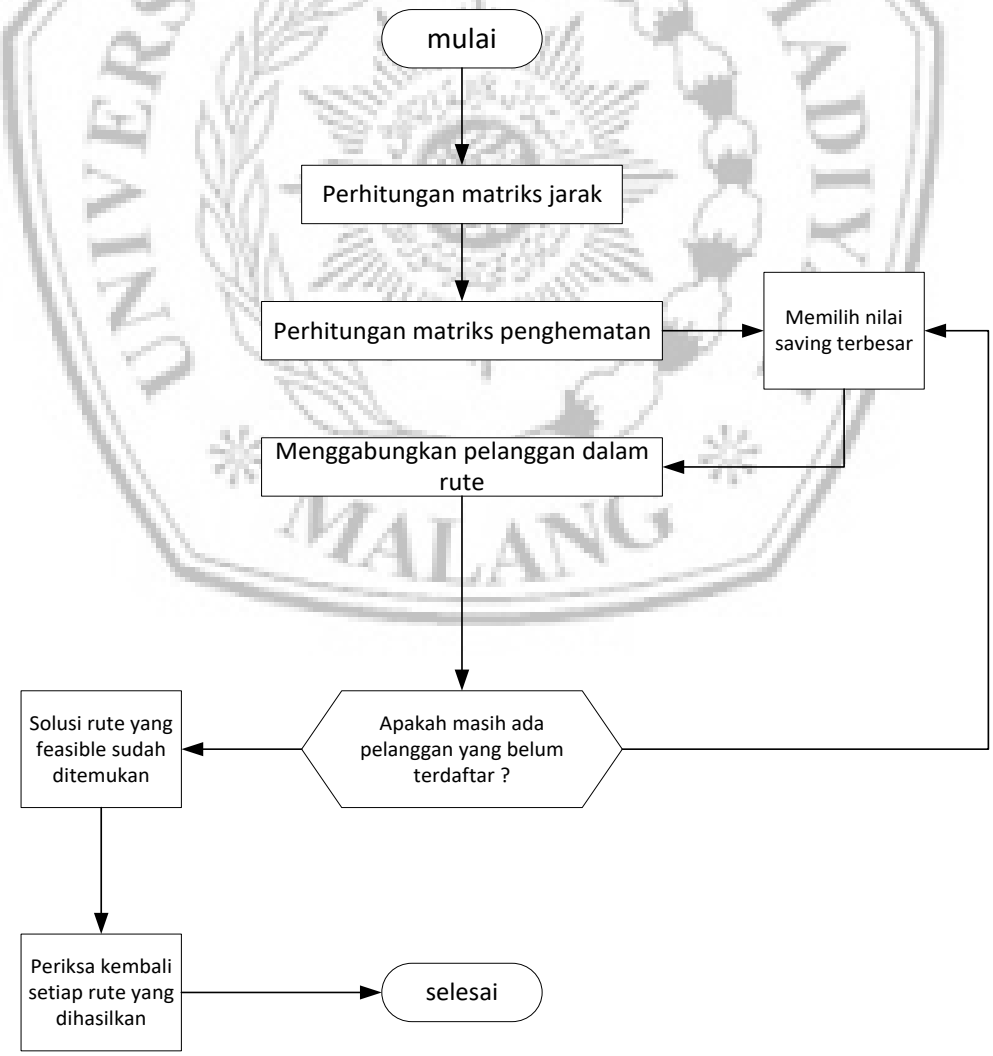
Ekivalen dengan biaya kendaraan Db pada Gambar 2.1 B adalah:

Db = c0i + cij + cj0

Dengan menggabungkan kedua rute memperoleh penghematan Sij:

Sij = Da – Db = ci0 + c0j – cij

Besarnya nilai Sij mengindikasikan suatu hal yang menarik, dengan biaya yang telah ditentukan, untuk mengunjungi titik i dan j pada rute yang sama di mana titik j dikunjungi setelah mengunjungi titik i. Ada 2 versi pada algoritma penghematan, versi berurutan (rentetan) dan versi paralel. Pada versi rentetan secara tepat, 1 rute dibuat/dijalani pada suatu waktu (tidak termasuk rute yang hanya dengan 1 pelanggan), sementara versi paralel lebih dari 1 rute dapat dijalani pada suatu waktu. Berikut adalah algoritma peghematan atau saving.



Gambar 2.2 *flowchart* standar algortma *saving matriks*

Langkah-langkah yang dilakukan dalam pengerjaan dengan menggunakan algoritma penghematan (Octora, 2014):

1. Langkah 1

   Inisialisasi data jarak, data jumlah permintaan, data waktu pelayanan, kecepatan rata-rata kendaraan dan kapasitas kendaraan sebagai input yang dibutuhkan, lanjut ke langkah 2.
2. Langkah 2

   Buat matriks jarak antar depot ke konsumen dan antar konsumen ke konsumen, lanjut ke langkah 3.
3. Langkah 3

   Hitung nilai saving menggunakan persamaan

   S(i,j) = d (D, i) + d (D ,j) – d (i,j)

   untuk setiap pelanggan untuk mengetahui nilai penghematan, lanjut ke langkah 4.
4. Langkah 4

   Urutkan pasangan pelanggan berdasarkan nilai saving matriks jarak dari nilai saving matriks terbesar hingga yang terkecil, lanjut ke langkah 5.
5. Langkah 5

   Pembentukan tur pertama (t=1), lanjut ke langkah 6.
6. Langkah 6

   Tentukan pelanggan pertama yang ditugaskan pada tur dengan cara memilih kombinasi pelanggan dengan nilai saving terbesar, lanjut ke langkah 7.
7. Langkah 7

   Hitung banyaknya jumlah permintaan dari konsumen yang telah terpilih. Apabila jumlah permintaan masih memenuhi kapasitas kendaraan maka lanjut ke langkah 8. Apabila jumlah permintaan melebihi kapasitas kendaraan maka dilanjutkan ke langkah 10.
8. Langkah 8

Hitung total jarak, waktu perjalanan, waktu pelayanan, dan total waktu berdasarkan pelanggan yang telah terpilih, lanjut ke langkah selanjutnya.

9. Langkah 9

   Pilih pelanggan selanjutnya yang akan ditugaskan berdasarkan kombinasi pelanggan terakhir yang terpilih dengan nilai saving terbesar, kembali ke langkah 7.

10. Langkah 10

    Hapus pelanggan terakhir yang terpilih, lanjut ke langkah 11

11. Langkah 11

    Masukkan pelanggan yang terpilih sebelumnya untuk ditugaskan kedalam tur maka tur (t) telah terbentuk. Apabila masih ada pelanggan yang belum terpilih maka lanjut ke langkah 12. Apabila semua pelanggan telah ditugaskan maka proses pengerjaan Algoritma *Clarke & Wright Savings* telah selesai.

12. Langkah 12

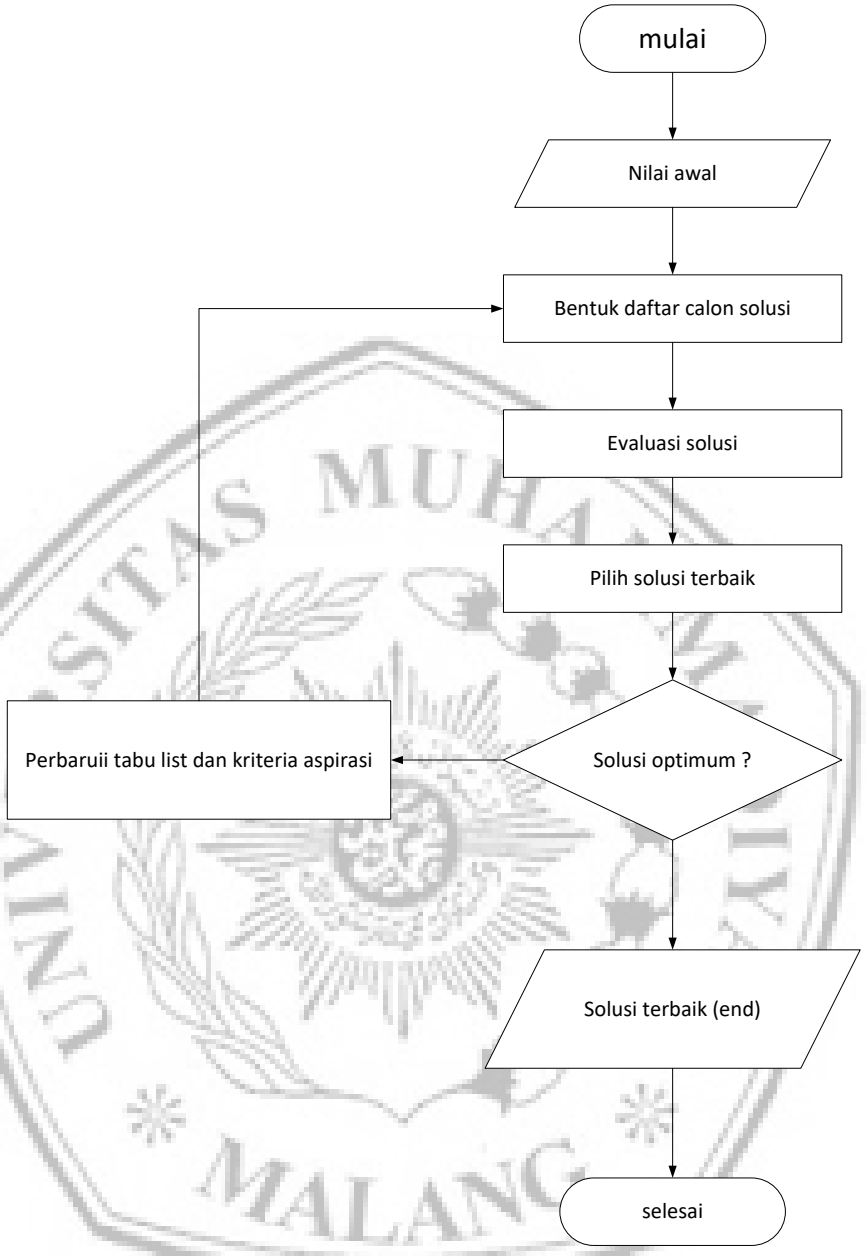
    Pembentukan tur baru(t = t+1), lanjut ke langkah 6

2.6. Algoritma *Tabu Search (TS)*

Menurut Sitorus (2014) *Tabu Search* merupakan suatu metode pencarian lokal iteratif yang biasa digunakan dalam optimasi kombinatorial. Tongan, suatu Bahasa Polinesia yang digunakan oleh suku Aborigin pulau Tonga untuk mengindikasikan suatu hal yang tidak boleh "disentuh" karena kesakralannya Konsep *Tabu Search* yang digunakan dalam penelitian ini diadopsi dari Glover dan Laguna (1997), yang terdiri atas tiga tahap: pencarian awal *(preliminary search)*, intensifikasi dan diversifikasi. Pada tahap pencarian awal, algoritma Tabu Search menyerupai metode optimasi yang lain, di mana perbedaan utamanya hanyalah bahwa pada *Tabu Search* sebuah solusi dapat diterima meskipun kualitas dari solusi tersebut tidak lebih baik daripada solusi awal. Pada tahap intensifikasi dilakukan pergerakan atau pencarian solusi di area sekitar solusi yang telah ditemukan pada

tahap pencarian awal, sedangkan tahap diversifikasi mencari solusi pada area-area baru *(eksplorasi).*

Istilah ‘tabu’ dalam *Tabu Search* berasal dari usaha untuk menghindari terjadinya *cycling* (kembali ke solusi awal), di mana beberapa pergerakan dinyatakan tabu atau tidak boleh dilakukan selama beberapa iterasi. Gerakan-gerakan yang tabu ini disimpan dalam daftar tabu atau *tabu list*. Dengan menggunakan *tabu list*, solusi yang tidak lebih baik dari solusi awal dapat diterima agar dapat menghindari kondisi terjebak dalam optimal lokal. Akan tetapi, dalam *Tabu Search* terdapat pula kondisi aspirasi *(aspiration condition)*, di mana jika solusi dari gerakan yang terdapat pada *tabu list* merupakan solusi yang lebih baik dari semua solusi yang telah ditemukan, maka solusi ini akan diterima dan dibebaskan dari *tabu list*. Berikut ini merupakan flowchart *tabu search* standar (Pradhana, 2011):



Gambar 2.3 *flow chart tabu search*

Menurut Pradhana (2011) Algoritma *tabu search* memiliki enam elemen utama yang digunakan untuk menyelesaikan VRP yaitu:

1. *Representasi* Solusi
2. Pembentukan Solusi Awal (*Initial Solution*)
3. Solusi *Neighborhood*
4. *Tabu list*

5. Kriteria Aspirasi
6. Kriteria Pemberhentian

2.6.1 Representasi Solusi

Representasi solusi yang digunakan dalam algoritma *tabu search* untuk VRP dalam tugas akhir ini adalah solusi *fisibel* yang ditulis sebagai suatu barisan titik-titik *(array)* dengan tiap titik tampak hanya satu kali dalam urutan. Titik yang dimaksud menunjukkan dealer (subdistributor). Untuk titik yang pertama dan terakhir adalah titik atau dealer 1 (depot), meskipun dalam representasi solusi titik 1 tidak harus selalu berada pada barisan pertama ataupun terakhir, karena rute yang terbentuk adalah suatu *cycle* (Pradhana, 2011).

2.6.2 Pembentukan Solusi Awal (*Initial Solution*)

Langkah awal yang dilakukan untuk menyelesaikan VRP menggunakan algoritma TS adalah membentuk solusi awal. Setiap solusi atau rute ditulis dalam bentuk array satu dimensi. Solusi awal dibentuk secara heuristik dengan mencari titik yang terdekat dengan depot dan menambahkan titik pada rute sepanjang tidak membentuk *cycle*, begitu seterusnya hingga semua titik dikunjungi (Pradhana, 2011).

2.6.3 Solusi Neighborhood

Dalam pencarian dengan teknik ini, setiap kemungkinan atribut dari struktur dapat dipindah-pindah. Perubahan yang dipakai oleh dua neighbourhood dengan melakukan swap elemen matriks atau kombinasi elemen itu dengan menukar elemen lain dalam *matriks*. Misalkan X adalah himpunan semua solusi, X, NX adalah himpunan *neighbourhood* solusi. Proses pencarian solusi bergerak dari solusi satu ke solusi selanjutnya dengan cara memilih solusi dalam solusi neighbourhood yang sudah ada yang tidak tergolong solusi terlarang VN untuk setiap iterasi. Solusi neighbourhood dalam tugas akhir ini didefinisikan sebagai solusi alternatif yang diperoleh dengan melakukan *move atau swap*. Solusi neighbourhood diperoleh dengan menukarkan dua titik yang berada dalam solusi.

Hal ini menjamin bahwa solusi yang terbentuk adalah solusi fisibel (Pradhana, 2011).

2.6.4 *Tabu list*

Untuk menghindari terulangnya langkah yang diambil, maka dilakukan tabu test dengan menggunakan *tabu list* yang sudah ada. *Tabu list* berisi atribut solusi-solusi yang telah dikunjungi sebelumnya. Tujuan utama dari *tabu list* bukan untuk mencegah terulangnya langkah yang telah diambil, namun lebih kepada agar tidak berjalan mundur. Situasi perulangan jarang sekali terjadi karena telah dikombinasikan dengan beberapa *neighbourhood* sehingga kemungkinan perulangan solusi yang telah dikunjungi hampir tidak mungkin.

Tabu Search menggunakan *tabu list* untuk menolak solusi-solusi yang memenuhi atribut untuk mencegah terjadinya *cycling* dalam proses pencarian solusi pada daerah yang sama dan menuntun proses pencarian untuk menelusuri daerah solusi yang berikutnya.

Ukuran *tabu list* untuk menghasilkan kualitas solusi yang baik akan bertambah seiring dengan membesarnya ukuran masalah. Namun, tidak ada aturan baku untuk menentukan ukuran *tabu list*. Hal ini disebabkan ukuran *tabu list* bergantung pada ketatnya kriteria tabu yang diterapkan. Ukuran *tabu list* yang terlalu panjang akan mengakibatkan buruknya kualitas solusi karena terlalu banyak *move* yang dilarang. Ukuran *tabu list* dalam tugas akhir ini diatur sedemikian rupa sehingga panjangnya sama dengan banyaknya iterasi yang telah ditetapkan sebelumnya (Pradhana, 2011).

2.6.5 Kriteria Aspirasi

Walaupun mempunyai peran sentral dalam TS, status tabu terkadang sangat kuat antara lain, tabu dapat melarang move yang atraktif, bahkan ketika tidak terdapat bahaya *cycling* atau status tabu mungkin mengarah pada stagnasi dalam

proses pencarian. Oleh karena itu, diperlukan suatu metode untuk membatalkan status tabu tersebut yang disebut *aspiration.*

Aturan dasar yang digunakan dalam kriteria aspirasi pada algoritma TS dalam tugas akhir ini adalah kualitas solusi terbaik dalam *neigbourhood* dan solusi yang terbentuk tidak sama dengan solusi yang sudah ada. Jika kualitas solusi baru dalam neigbourhood lebih baik dibanding dengan yang dicapai sebelumnya, maka solusi baru tersebut dicatat pada list sebagai solusi terbaik yang baru dan status tabu dicabut. Apabila kualitas solusi baru tidak lebih baik dari solusi sebelumnya maka solusi tersebut tetap dimasukkan dalam list tetapi status tabu tetap berlaku pada solusi tersebut. Kemudian proses pencarian dilanjutkan sampai kriteria pemberhentian *(termination criteria)* dipenuhi (Pradhana, 2011).

2.6.6 Kriteria Pemberhentian

Dalam teori, pencarian dapat dilakukan terus kecuali bila nilai optimal dari masalah yang diselesaikan sudah diketahui sebelumnya. Pada praktiknya pencarian dihentikan dengan sungguh-sungguh pada beberapa titik. Kriteria pemberhentian yang biasa digunakan dalam TS adalah.

1) setelah semua iterasi yang telah ditetapkan sebelumnya terpenuhi;
2) setelah beberapa iterasi tanpa ada perbaikan pada nilai fungsi objektif;
3) ketika fungsi objektif mencapai nilai batas atas atau nilai batas bawah yang telah ditentukan sebelumnya;
4) ketika tidak ada lagi solusi baru yang dapat dibangkitkan dari *current neighbourhood solution* dimana semua move terdapat dalam *tabu list*.

Kriteria pemberhentian *(termination criteria)* yang dipakai dalam tugas akhir ini yaitu setelah semua iterasi yang telah ditentukan terpenuhi. Jumlah iterasi yang dipilih yaitu sama dengan banyaknya dealer karena jumlah iterasi maksimal sama dengan panjang *tabu list* (Pradhana, 2011).

2.7. Langkah-langkah Algoritma TS

1. Menentukan solusi awal dan menetapkannya sebagai solusi optimum.
2. Menentukan solusi alternatif yaitu dengan melakukan *move* (menukarkan) dua titik dalam solusi.
3. Mengevalusi solusi-solusi alternatif dengan *tabu list* untuk melihat apakah kandidat solusi (solusi alternatif) tersebut sudah ada pada *tabu list*. Apabila solusi alternatif sudah ada dalam *tabu list*, maka solusi alternatif tersebut tidak akan dievaluasi lagi. Apabila solusi alternatif belum terdapat dalam *tabu list*, maka solusi alternatif tersebut disimpan dalam *tabu list* sebagai solusi alternatif terbaik.
4. Memilih solusi terbaik dan menetapkannya sebagai solusi optimum baru.
5. Memperbarui *tabu list* dengan memasukkan solusi optimum baru.
6. Apabila kriteria pemberhentian terpenuhi maka proses berhenti dan diperoleh Solusi optimum.
7. Jika tidak, proses kembali berulang dimulai dari langkah ke dua


UNIVERSITAS MUHAMMADIYAH
MALANG